DAS: Density-Aware Sliders

Name	Name	Name
Affiliation	Affiliation	Affiliation
Address	Address	Address
Email	Email	Email
Phone	Phone	Phone

ABSTRACT

We present a selection widget that utilizes pre-existing knowledge of the distribution of the underlying data to ease browsing and selection. Standard sliders suffer from two problems: poor subpixel data querying, and uniform visual representation of non-uniform data. We provide density-aware sliders to give users more efficient subpixel data querying and use embedded visualizations to introduce or improve the visual representation of the data contained by the slider. Through a controlled user study, we find that our proposed density-aware slider outperforms the standard Alphaslider with large datasets.

Author Keywords

Dynamic query, information visualization, slider, data selection, Alphaslider, lasso, range slider, density-aware

ACM Classification Keywords

H5.2. Information Interfaces and presentation (e.g., HCI): User Interfaces.

INTRODUCTION

Dynamic queries are powerful and effective interactive components for filtering information from large datasets [5]. They facilitate rapid, incremental and reversible actions for identifying trends and outliers in data. Several interface widgets make available the advantages of dynamic queries, such as sliders, checkboxes and buttons.

Sliders are common data filtering widgets. They however lack an awareness of their underlying data distribution, and have limited visual feedback of the data density. These problems are especially important when the interface maps more than one item to any pixel along the slider track, also referred to as subpixel mapping or querying facility. Subpixel mapping problems with sliders are particularly evident with the Alphaslider, which is designed to query large sorted lists of alphanumeric items [1].

New widgets have been proposed to improve slider interaction [2] [3]. The PVSlider uses a popup vernier to give users sub-pixel-pitch control and sub-pixel visualization. The FineSlider uses an elastic band metaphor to give users more intuitive control over scrolling speed and precision where a longer elastic band correlates to faster scrolling and a shorter band correlates to more precise and deliberate item selection. The Zlider, proposed by Ramos et al, takes advantage of pressure input in a pressure sensitive environment to fluidly change from coarse to fine granularity and shift granularity control from the system to the user [4]. None of the above sliders dynamically adapt to the data they contain. Ideally, mechanics of the slider should adjust according to the distribution of the data contained by the slider. We believe that dynamic mechanics improve over static mechanics because they can provide the best possible experience in different situations while static mechanics provide and experience that might excel in some cases and perform poorly in others.

We propose density-aware sliders (DAS) that slider widgets informed of the underlying data content they query to simplify data filtering. Our sliders intelligently redistribute the items in densely packed pixels over a larger area providing users with a visualization that gives them a sense of location in the pixel (Figure 2). This information lets the user better decide when to change querying granularity. We also equip our data-aware sliders with embedded visualizations of how the data is distributed, similar to scented widgets [1]. Users can modify their behavior by

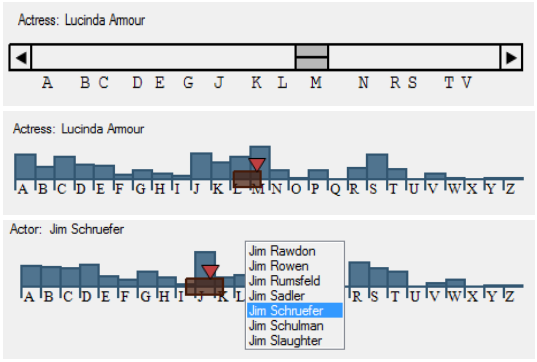

Figure 1: (Top) Display Distortion with the Alphaslider. Letter spacing indicates distribution and users have constant input. (Middle) Input Distortion with the ActiveArea Slider. Histograms indicate distribution and user input gets distorted based on local density. (Bottom) The ActiveList Slider with an Input Distortion.

Comment [i1]: Why do we believe density aware is good, need to state this.

Does the following sentence do an adequate job answering the question?

Submitted to CHI 2013 – this had the copyright info previously. Should I make this textbox smaller or keep it the same size for when the copyright info is put back in?

leveraging the density and distribution information provided by our new visualization.

In a comparative evaluation, we find that participants can filter large datasets 15.5% faster with DAS than the Alphaslider. Moreover, user preferred our design of DAS to current methods. Our contribution is in demonstrating the value of a new dynamic query slider based on the distribution of its underlying data set.

DESIGNING OF MULTI-ITEM PIXEL ASSIGNMENT

Mapping multiple items to a single pixel and giving users quick access to each item can lead to inefficient exploration of large data sets. The Alphaslider mitigates the multi-pixel mapping through coarse or fine-grained dragging of the slider thumb. Coarse dragging lets the user skip through the 10 items at a time while fine-grained dragging lets the user move through the list one item at a time. This however can be inefficient. Furthermore, the Alphaslider's querying method gives no visual feedback when the number of items per pixel exceeds the coarse-grained movement value making highly dense data even more difficult to navigate.

Subpixel visual feedback is also an issue intricately associated with the Alphaslider. This problem was identified by Ayatsuka et al [2] who used a popup vernier to provide user with a subpixel visualization. However, their solution lacks the ability to adapt dynamically to the contained data. This requires the application designer to know exactly how many items will be contained by and the size of the slider to provide a popup vernier that accurately represents the distribution and density of items.

We propose a multilevel approach for addressing the sub-pixel mapping problem. Our solution solves the issue of slider weight by providing users with pixel based instead of list based coarse querying. Pixel based querying also avoids the issue of receiving no visual feedback during coarse-grained movement in sufficiently large datasets. A multilevel approach improves subpixel querying by mapping the items in a single pixel along the width of slider (items k to k + m in figure 2). Users can individually select items r to r + p. When p > 0, more than one item is mapped to each horizontal pixel of the main slider. We believe a list that enumerates items r to r + p for p > 0 is highly beneficial for quick access and selection.

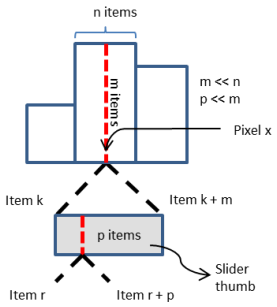

Figure 2: Visual representation of our multilevel approach to subpixel querying. The center histogram contains n items. Pixel x contains m items which are mapped across the slider thumb. Each horizontal pixel of the thumb contains p items.

Our multilevel approach behaves differently depending on the distortion style. An input distortion causes the items in any given pixel to vary (m is not constant). This means that the number of items mapped along the slider varies which forces the user to distort their input when they navigate from one index to the next. The display distortion keeps m constant across the entire width of the slider track allowing users to have constant input for all indices.

EXPERIMENTAL DESIGN

We conducted an experiment to compare different designs of density-aware sliders against the Alphaslider. We chose to compare our sliders against the Alphaslider instead of the FineSlider because [2] showed that the FineSlider performs worse and is less favored. We did not choose the PVSlider because our dataset was generated randomly at runtime.

Apparatus

We built the interfaces for the experiment using Visual Studio. Participants used a standard 3-button mouse and a 23.5" Dell monitor with a resolution of 1920x1080 pixels. The experimental system consisted of an Intel i5-2400 CPU with a clock speed of 3.1 GHz and 4 GB of RAM.

Participants

Twelve participants completed four trials for each condition. Participants ranged from 20 to 39 years of age.

Interfaces

The experiment used three different sliders and two distortion styles (Figure 1).

Alphaslider

The Alphaslider (Figure 1 (top)) has four navigation techniques. Users can jump directly to an item in the list by clicking anywhere in the bounded area above the letters. Users can navigate through the list at a rate of ten items per mouse movement by clicking in the top tile of the slider thumb and dragging. Users can also navigate one item at a time either by clicking on the arrows at the ends of the slider or by clicking in the bottom tile of the slider thumb and dragging.

ActiveArea Slider

The ActiveArea Slider (Figure 1 (middle)) allows users to navigate by clicking on and dragging the slider thumb, by clicking on and dragging the secondary red slider or by pressing the left or right arrow keys on the keyboard. The main slider has a variable width that changes based on the density of the information of the pixel it queries at any given point. The system maps a maximum of four items to each horizontal pixel of the slider. Users can drag the red triangle (a secondary slider) to query items mapped to the main slider. Users can also roll the mouse wheel to navigate one item at a time. Rolling the mouse wheel appropriately moves the secondary slider and gives users continuous querying whereas users cannot drag the secondary slider past either edge of the main slider.

ActiveList Slider

The ActiveList Slider (Figure 1 (bottom)) incorporates a list into the ActiveArea Slider. The queried subpixel item determines the contents of the list. The items of the list are the items r through r + p. If p is less than a minimum threshold of six then the list is enumerated up until that threshold is met or until item k + m is shown. Dragging the red slider and rolling the mouse wheel appropriately updates the list. The edges of the main slider provide physical limits to querying by dragging the secondary slider while the mousewheel lets users perform continuous querying.

Input Distortion

This distortion (Figure 1 (middle and bottom)) uses histograms to give users information about item distribution. We call this Input Distortion because users have to distort their input based on the local density.

Display Distortion

This distortion (Figure 1 (top)) uses variable spacing between letters to give users information about item distribution. Users do not have to distort their input because item density is uniform.

Hypotheses

This paper is primarily concerned with design of an efficient density-aware slider. The speed with which a user locates an item is largely dependent on total mouse movement. For each of the sliders there is a period of coarse querying where the user approximates the location of the target and a period of querying where the user does fine adjustments to acquire the target. Based on this assumption we have the following hypotheses:

H1) The ActiveList slider will perform the best because it does the best job breaking the query into multiple levels.

H2) All of our sliders will outperform the Alphaslider because they give the user better control and visual feedback.

H3) The ActiveList Slider will be preferred because the technique gives users better, quicker and easier access to the data.

Design and Procedure

The experiment used a 3x2x2 within-subject design. The independent variables were:

1. Technique (*Alphaslider (AS)*, *ActiveArea Slider (AAS)* or *ActiveList Slider (ALS)*)
2. Distortion Style (*Input Distortion (ID)* or *Display Distortion (DD)*)
3. Local Density (*Low* or *High*)

We measured target acquisition time and error rate. We also collected participants' preferences.

Task

The system provided users with a search for each condition. Each condition had four trials, which the users completed consecutively. The search target would be a name that the users had to find using the appropriate slider. Upon completion the task users would press the spacebar and be given a new task. With each new task, the slider thumb returned to the beginning of the track.

Procedure

We first ran a pilot study to filter poorly designed and inefficient interfaces. Prior to the timed trials the participants were given a set of tasks under each condition (the ordering of which was determined by a balanced Latin square) to familiarize themselves with each interface while reading interface specific instructions and ask questions. During the timed trial, the experimenters disallowed participants from asking questions and required them to fill out a NASA TLX form for each technique and distortion style. Upon completion of the experiment, the participants filled out forced pairwise comparison sheet on which they indicated their preferred interface for every combination of technique and distortion style. The experimenters also asked participants to write down any comments they have regarding the experiment, techniques, and distortion styles.

Comment [i2]: Are users distorting their input, or is the system doing this?

The users.

Comment [i3]: Looks like Fig 1, top is the alpha slider. Check to see if this is correct?

It is.

RESULTS

We used a univariate general linear model for our ANOVA analysis of acquisition time and error rate. We performed a Bonferroni post-hoc pairwise comparison (equal (?) variances) on the acquisition time data. We used Friedman's X^2 test to analyze the TLX data.

Comment [p4]: I don't know what equal and inequal variances means.

Acquisition Time: Univariate ANOVA reveals significant main effect of technique ($F_{(2, 22)} = 7.548$, $p = 0.003$) and of local density ($F_{(1, 11)} = 5.432$, $p = 0.04$). The ANOVA analysis also reveals significant interaction effects between local density and distortion style ($F_{(1, 11)} = 16.672$, $p = 0.02$). We believe this interaction effect arises because of the fact that the display distortion equalizes the density between indices. Post-hoc pairwise analysis showed significance between ALS and AS ($p < 0.001$) and between ALS and AAS ($p = 0.001$).

Error Rate: We found no significant difference in error rate among techniques ($F_{(2, 22)} = 1.000$, $p = 0.384$) or among distortion styles ($F_{(1, 11)} = 0.000$, $p = 1.000$). Participants were 98% accurate with the Alphaslider and 99% accurate with the ActiveArea Slider and the ActiveList Slider. Participants were also 99% with both distortion styles and both densities.

TLX Analysis: Analysis of the TLX information indicates that there were significant main effects for frustration and effort among interfaces (X^2 (2) = 7.386, $p < 0.05$ and X^2 (1) = 18.286, $p < 0.001$, respectively). No significant main effects were found for distortion style for either frustration or effort (X^2 (2) = 0.034, $p = 0.853$ and X^2 (1) = 0.533, $p = 0.465$, respectively). Significant interaction effects were

found for effort (X^2 (5) = 22.134, $p < 0.001$) and for frustration (X^2 (5) = 12.481, $p < 0.03$). Post-hoc analysis with Wilcoxon Signed-Rank Tests and Bonferroni corrections revealed statistically significant differences in effort between the ActiveList Slider with both distortions and every other condition ($p < 0.01$). The post-hoc analysis also revealed no significant differences for frustration. This information supports H3.

DISCUSSION

Our results show statistically significant differences between the techniques. Specifically, a difference is found between the ActiveList Slider and Alphaslider and the ActiveList Slider and ActiveArea Slider ($p <= 0.001$). Our results do not show significant differences between distortion styles.

Overall, participants preferred our sliders to the Alphaslider. This is because the Alphaslider requires that users have fine motor skills to query one item at a time or to click on arrow buttons repeatedly while our interfaces let users use the mousewheel, which is highly precise and requires less motor skills without sacrificing speed. Many users commented, "I really like using the mousewheel" and even sometimes instinctively attempted using the mousewheel while querying with the Alphaslider. One user sighed, "Oh good, the one with the list" in relief during the experiment.

Lessons Learned

We learned that while users prefer using the input distortion style they performed marginally but insignificantly better with it. We believe that the added complexity of requiring variable input offsets the visual advantage it provides over the display distortion. We also learned that users heavily prefer using a mousewheel for fine-grained querying to cursor movement. We believe that this is because of the higher motor precision required to move the cursor.

Limitations

We suspect that with ultra-dense datasets (greater than 100 000 items) we will start seeing performance degradation with our sliders because of the instinctive behavior of users. Ideally, users should use the different granularities provided by our sliders: the main thumb, the secondary thumb, and the mousewheel. Users tend not to use the secondary thumb because they perceive it as having the same or similar granularity as the mousewheel, which is true for small to medium density datasets but not for larger datasets. At approximately 20 000 items the granularity difference between the mousewheel and secondary thumb starts becoming noticeable. We believe that explicit training or a new visual cue can solve this problem.

Applications

One major application of our work can be in video editing where editors need to select from tens or hundreds of thousands of frames to edit. Our sliders can give editors highly precise frame selection while still adapting to movies of varying length or movies with variable frame rates. Another possible application lies in the world of math with precise visual integration. Our sliders can allow users to set upper and lower bounds for visual integration while also giving users the flexibility to adjust the precision on either end of the integral independently while receiving appropriate visual feedback.

CONCLUSIONS AND FUTURE WORK

We presented density-aware sliders and a new slider distortion style that were meant to reduce querying time while keeping user error rates constant compared to the Alphaslider. We found that our new distortion style was ineffective but that our proposed density-aware sliders show promise for large datasets. We plan explore under what conditions our density-aware sliders excel. In the future, we would also like to investigate other possibilities for density-aware widgets such as a range slider or lasso selection tool.

AKNOWLEDGEMENTS

We thank NSERC for funding part of this research and the participants for the valuable time and effort.

REFERENCES

1 Ahlberg, Christopher and Shneiderman, Ben. The Alphaslider: A Compact and Rapid Selector. In *Human Factors in Computing Systems* (Boston 1994), 365-371.

2 Ayatsuka, Yuji, Rekimoto, Jun, and Matsuoka, Satoshi. Popup Vernier: a Tool for Sub-pixel-pitch Dragging with Smooth Mode Transition. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (1998), 39-48.

3 Masui, Toshiyuki, Kashiwagi, Kouichi, and Borden, George R. IV. Elastic Graphical Interfaces to Precise Data Manipulation. In *CHI Conference Companion on Human Factors in Computing Systems* (1995), 143-144.

4 Ramos, Gonzalo and Balakrishnan, Ravin. Zliding: Fluid Zooming and Sliding for High Precision Parameter Manipulation. In *Proceedings of UIST* (2005), 143-152.

5 Shneiderman, Ben. Dynamic Queries for Visual Information Seeking. *IEEE Software*, 11, 6 (November 1994), 70-77.

6 Willett, Wesley, Heer, Jeffrey, and Agrawala, Maneesh. Scented Widgets: Improving Navigation Cues with Embedded Visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13, 6 (November 2007), 1129-1136.

Comment [P5]: Should I include F?

Comment [i7]: These should be sorted alphabetically with last name first.

Comment [P6]: Not entirely sure what to put here.

The columns on the last page should be of approximately equal length.
Remove these two lines from your final version.